11/06/00

A

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

CERTIFICATE OF EXPRESS MAILING

I hereby certify that this paper and the documents and/or fees referred to as attached herein are being deposited with the United States Postal Service on November 2, 2000 in an envelope as "Express Mail Post Office to Addressee" service under 37 CFR §1.10, Mailing Label Number **EL610161542US**, addressed to the Assistant Commissioner for Patents, Washington, DC 20231.

Joe Brock

Attorney Docket No. INSTP007B

First Named Inventor: Mark W. Bradley

UTILITY PATENT APPLICATION TRANSMITTAL (37 CFR § 1.53(b))

Assistant Commissioner for Patents
Box Patent Application
Washington, DC 20231

☐ Duplicate for fee processing

Sir: This is a request for filing a patent application under 37 CFR § 1.53(b) in the name of inventor:
Mark W. Bradley

For: MODULAR SOFTWARE METHOD FOR INDEPENDENT STORAGE NODES

Application Elements:

- ☒ 31 Pages of Specification, Claims and Abstract
- ☒ 12 Sheets of Drawings (Informal)
- ☒ 02 Pages Combined Declaration and Power of Attorney

Accompanying Application Parts:

- ☒ Assignment and Assignment Recordation Cover Sheet (recording fee of **$40.00** enclosed)
- ☐ 37 CFR 3.73(b) Statement by Assignee
- ☐ Information Disclosure Statement with Form PTO-1449
  - ☐ Copies of IDS Citations
- ☐ Preliminary Amendment
- ☒ Return Receipt Postcard
- ☒ Small Entity Statement(s)
- ☐ Other:

Fee Calculation (37 CFR § 1.16)

	(Col. 1) NO. FILED	(Col. 2) NO. EXTRA	SMALL ENTITY RATE	FEE	OR	LARGE ENTITY RATE	FEE
BASIC FEE			$355	$355.00	OR	$710	$
TOTAL CLAIMS	08	-20 = 00	x09=	$	OR	x18 =	$
INDEP CLAIMS	02	-03 = 00	x40 =	$	OR	x80 =	$
[ ] Multiple Dependent Claim Presented			$135 =	$	OR	$270 =	$
* If the difference in Col. 1 is less than zero, enter "0" in Col. 2.			Total	**$355.00**	OR	Total	$

☒ Check No. 4454 in the amount of **$395.00** is enclosed.

☒ The Commissioner is authorized to charge any fees beyond the amount enclosed which may be required, or to credit any overpayment, to Deposit Account No. 50-0805 (Order No. INSTP007B).

General Authorization for Petition for Extension of Time (37 CFR §1.136)

☒ Applicants hereby make and generally authorize any Petitions for Extensions of Time as may be needed for any subsequent filings. The Commissioner is also authorized to charge any extension fees under 37 CFR §1.17 as may be needed to Deposit Account No. 50-0805 (Order No. INSTP007B).

☒ Please send correspondence to the following address:

Joe A. Brock
MARTINE PENILLA & KIM, LLP
710 Lakeway Drive, Suite 170
Sunnyvale, CA 94085
Tel (408) 749-6900
Fax (408) 749-6901

Date: November 2, 2000

Joe A. Brock II, Esq.
Registration No. 46,021

Applicant/Patentee: Independent Storage Corp.

Application or Patent No. _____ Atty Docket # INSTP007B

Filed or Issued: Not filed

VERIFIED STATEMENT (DECLARATION) CLAIMING SMALL ENTITY STATUS
37 CFR 1.9(f) and 1.27(c)--SMALL BUSINESS CONCERN

I hereby declare that I am
[ ] the owner of the small business concern identified below:
[X] an official empowered to act on behalf of the small business concern identified below:

NAME OF CONCERN: Independent Storage Corp.

ADDRESS: 7101 La Vista Place, Niwot, CO 80503

I hereby declare that the above identified small business concern qualifies as a small business concern as defined in 13 CFR 121.3-18, and reproduced in 37 CFR 1.9(d), for purposes of paying reduced fees under 41(a) and (b) of Title 35, U.S. Code, in that the number of employees of the concern, including those of its affiliates, does not exceed 500 persons. For purposes of this statement, (1) the number of employees of the business concern is the average over the previous fiscal year of the concern of the persons employed on a full-time, part-time or temporary basis during each of the pay periods of the fiscal year, and (2) concerns are affiliates of each other when either, directly or indirectly, one concern controls or has the power to control the other, or a third party or parties controls or has the power to control both.

I hereby declare that rights under contract or law have been conveyed to and remain with the small business concern identified above with regard to the invention entitled: MODULAR SOFTWARE METHOD FOR INDEPENDENT STORAGE NODES, by inventor Mark W. Bradley, described in

[X] the specification filed herewith.
[ ] Application No. _____ filed _____
[ ] patent # _____ issued _____

If the rights held by the above-identified small business concern are not exclusive, each individual, concern or organization having rights to the invention is listed below* and no rights to the invention are held by any person, other than the inventor, who could not qualify as a small business concern under 37 CFR 1.9(d) or by any concern which would not qualify as a small business concern under 37 CFR 1.9(d) or a nonprofit organization under 37 CFR 1.9(e).
*Note: separate verified statements are required from each named person, concern or organization having rights to the invention averring to their status as small entities. (37 CFR 1.27)
Name: _____
Address: _____
[ ] individual [ ] small business concern [ ] nonprofit organization

I acknowledge the duty to file, in this application or patent, notification of any change in status resulting in loss of entitlement to small entity status prior to paying, or at the time of paying, the earliest of the issue fee or any maintenance fee due after the date on which status as a small entity is no longer appropriate. (37 CFR 1.28(b)).

I hereby declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true; and further, that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under 1001 of Title 18 of the U.S. Code, and that such willful false statements may jeopardize the validity of the application, any patent issuing thereon, or any patent to which this verified statement is directed.

NAME OF PERSON SIGNING: Mark W. Bradley Mark W. Bradley

TITLE IN ORGANIZATION: President

ADDRESS OF PERSON SIGNING: 7101 La Vista Place, Niwot, CO 80503

ATTORNEY DOCKET No.
INSTP007B

PATENT APPLICATION

MODULAR SOFTWARE METHOD FOR INDEPENDENT STORAGE NODES

INVENTOR: Mark Bradley
1946 Lefthand Canyon Drive
Boulder, CO 80302
U.S. Citizen

ASSIGNEE: Independent Storage Corporation
7101 La Vista Place
Niwot, CO 80503

MARTINE PENILLA & KIM, LLP
710 Lakeway Drive, Suite 170
Sunnyvale, CA 94085
Telephone (408) 749-6900

MODULAR SOFTWARE METHOD FOR INDEPENDENT STORAGE NODES

by Inventor

Mark Bradley

CROSS REFERENCE TO RELATED APPLICATIONS

This application is related to the following applications: (1) U.S. Patent Application No. _____ (Attorney Docket No. INSTP001), filed September 13, 2000, and entitled "File Consistency Protocols and Methods for Carrying out the Protocols"; (2) U.S. Patent Application No. _____ (Attorney Docket No. INSTP002), filed on the same day as the instant application, and entitled "Dynamic Flat File Systems and Methods for Using the Same"; (3) U.S. Patent Application No. _____ (Attorney Docket No. INSTP005), filed October 6, 2000, and entitled "N-Way Data Mirroring System and Methods for Using The Same"; and (4) U.S. Patent Application No. _____ (Attorney Docket No. INSTP007A), filed October 31, 2000, and entitled "Independent Storage Architecture." Each of these related application is incorporated herein be reference.

BACKGROUND OF THE INVENTION

1. Field of the Invention

This invention relates generally to computer storage, and more particularly to computer storage systems and methods that are independent of microprocessor

architecture, microprocessor type, transport hardware or I/O device controller type that comprise an independent storage node.

2. Description of the Related Art

Today, more and more reliance is being placed on computers to create, edit, modify, and store important documents. With this reliance, comes the need for increased versatility in the ability to store and retrieve data. There are several techniques available today for facilitating computer storage, and as will be seen, more versatility in computer information storage is needed.

Figure 1A is a block diagram showing a conventional local storage system 100. The local storage system 100 includes a user computer 102 and a storage device 112 coupled to the user computer 102 via a peripheral interconnect 114. The user computer 102 has executed thereon an application 104, an operating system (O/S) having a local file system 106, a driver 108 for an input/output (I/O) device 110 that controls the storage device(s) 112.

To store and retrieve data for the storage device 112, the application 104 typically transmits a request to the local file system 106, which in turn passes the file system request to the device driver 108. The device driver 108 converts the file system request into a block-level I/O request that is then passed from I/O device 110 over the peripheral interconnect 114 to the storage device 112.

The I/O request then completes and the completion result is passed back up through the I/O device 110, driver 108, and local file system 106. The data is either

placed into system memory, for a disk READ operation, or placed on the storage device 112 for a disk WRITE command.

To provide increased flexibility the network file system (NFS) has been used to store data remotely from a user computer. Figure 1B is a block diagram showing a prior art network file system 150. The network file system 150 includes a user computer 102 and a storage computer 130 coupled to the user computer 102 via a network 116. Using the conventional NFS 150 a user can store data on a remote server 130 that is executing the same or a compatible O/S as the user computer 102.

To accomplish remote NFS storage, the user computer 102 includes an application 104, a network file system 120, protocols 122, a network driver 124, and a network interface card (NIC) 126 coupled to the network 116 via transport 128. The storage computer 130 includes a corresponding NIC 126, network driver 124, protocols 122, and file system 120. In addition, the server 130 includes a device driver 108, and an I/O device 110 coupled to a storage device 112 via a peripheral interconnect 114.

In the conventional NFS 150, the network file system 120 replaces the local file system 106 of Figure 1A, and is typically layered on TCP/IP or UDP/IP protocols 112. To store and retrieve data using NFS 150, the application 104 typically transmits a file system request to the network file system 120, through the protocol stacks 122 to the network driver 124. The network driver 124 then transmits the request from the NIC 126 of the user computer 102 to the NIC 126 of the storage computer 130 via the network 116 and transport connections 128.

When received by the NIC 126 on the storage computer 130, the request is passed through the network driver 124 and protocol stacks to the file system 120, which in turn passes the file system request to the device driver 108 of the storage computer 130. The device driver 108 then converts the file system request into a block-level I/O request that is passed from I/O device 110 to the storage device 112 via the peripheral interconnect 114.

When the I/O request completes, the result is passed back up through the I/O software layers of both the storage computer 130 and the user computer 102 using the network 116. The data is then either placed into the memory of the user computer for a READ operation, or placed on the storage device 112 for a WRITE command.

A problem with the conventional network file system 150 is that generally both the user computer 102 and the storage computer 130 need to execute the same O/S and file system 120. Thus, if the user computer 102 is executing on WINDOWS NT, the storage computer 130 also needs to execute WINDOWS NT for the network file storage system 150 to operate properly.

Thus, if the storage computer 130 is used with a plurality of user computers 102, all the user computers 102 generally must execute the same O/S and file system 120 as is executed on the storage computer 130. However, it is not always desirable to use a particular O/S for a particular application. Hence, different users often operate under different O/S's, and therefore may not be able to share the storage system 150 of a remote computer in this way.

Moreover, in both the convention local storage system 100 and NFS 150, the software controlling the storage aspects of the systems generally must be specifically written to support the specific hardware comprising the system, including the microprocessor architecture, which includes the endian-ness, internal design, and internal bit architecture of the processor. This limitation causes problems when hardware within a conventional storage system is changed, since the original software may no longer execute properly on the system. Thus, whenever new devices, processors, or other system hardware is changed, the system software of a conventional local storage system 100 or NFS 150 generally must be completely rewritten to function with the new system hardware.

In view of the forgoing, there is a need for a storage system that is capable of operation independent of the operating systems' limitations and which allows the storage not to be hosted by another computer system. The storage system should be capable of storing data remotely and capable of operating independently of any particular consumer computers' O/S. Further, the storage system should be capable of operating with reduced user configuration or networking knowledge, such that the user generally does not need to learn how to use a configure complex networked file and storage systems. In order to provide this functionality, storage nodes should be configurable to support many differing transport drivers, transport protocols, I/O device drivers and varying hardware configurations.

SUMMARY OF THE INVENTION

Broadly speaking, the present invention fills the needs described above by providing a storage system capable of providing data storage independent of a consumer computer architecture, independent of the transport, and independent of various network operating system environments. To this end, the storage nodes of the system are preferably embodied in a hardware component, or assembly, and a software component comprising a number of software modules providing the aforementioned functionality. The software modules themselves include well defined messaging interfaces that enable the modules to be combined, added to, or removed in order to provide the necessary functionality in the storage node. Further, the software is preferably capable of running on any microprocessor, regardless of the microprocessor architecture without modification of the source code.

In one embodiment, a system for modular storage software is disclosed. The system includes a module to module interface that is capable of receiving a message, which is configured in a first format, and translating the received message into a second format. In communication with the module interface is a first software module that is capable of sending messages configured in the first format to the module interface. Further, a second software module is in communication with the module interface. The second software module is capable of communicating messages configured in the second format to the module interface. In this manner, the first software module is capable of communicating with the second software module via the module interface to facilitate data storage and retrieval.

In another embodiment, an independent storage node is disclosed that includes a processor and transport hardware in communication with the processor that is capable of communicating data via a transport connection. Executing on the processor is modular storage software that comprises a plurality of software modules and a module interface. The module interface allows dynamic binding of the software modules and is capable of executing on a plurality of processor types by using particular software modules related to a specific processor type.

Advantageously, embodiments of the present invention can provide platform-independent storage for consumer computers, regardless of underlying hardware, software, including the operating system, protocols or physical transport. Other aspects and advantages of the invention will become apparent from the following detailed description, taken in conjunction with the accompanying drawings, illustrating by way of example the principles of the invention.

BRIEF DESCRIPTION OF THE DRAWINGS

The invention, together with further advantages thereof, may best be understood by reference to the following description taken in conjunction with the accompanying drawings in which:

Figure 1A is a block diagram showing a conventional local storage system;

Figure 1B is a block diagram showing a prior art network file storage system;

Figure 2 is a high-level illustration of an independent storage system, in accordance with an embodiment of the present invention;

Figure 3 is a block diagram showing a remote independent storage system, in accordance with an embodiment of the present invention;

Figure 4 is a block diagram showing an independent storage logical platform, in accordance with an embodiment of the present invention;

Figure 5 is a block diagram showing a communication channel, in accordance with an embodiment of the present invention;

Figure 6A is a block diagram showing an exemplary communication channel, in accordance with an embodiment of the present invention;

Figure 6B is a block diagram showing another exemplary communication channel, in accordance with an embodiment of the present invention;

Figure 7A is a block diagram showing an independent storage system supporting multiple file types, in accordance with an embodiment of the present invention;

Figure 7B is a block diagram showing an independent storage system supporting a single file type, in accordance with an embodiment of the present invention;

Figure 8 is a flowchart showing a method for providing data storage independent of an underlying architecture, in accordance with an embodiment of the present invention;

Figure 9 is block diagram showing a system for independent storage-to-storage communication, in accordance with an embodiment of the present invention; and

Figure 10 is a flowchart showing a method for providing independent storage using an independent storage node, in accordance with an embodiment of the present invention.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

An invention is disclosed for an independent storage system in a (distributed) computer environment. The present invention provides data storage independent from the particular operating systems executing on associated user computers. To this end, the storage nodes of the system are preferably embodied in a hardware component and a software component comprising a number of software modules, which include well defined messaging interfaces that enable the modules to be combined, added to, or removed. Using these software modules, embodiments of the present invention are capable of running on any microprocessor, regardless of the microprocessor architecture without modification of the source code.

In the following description, numerous specific details are set forth in order to provide a thorough understanding of the present invention. It will be apparent, however, to one skilled in the art that the present invention may be practiced without some or all of these specific details. In other instances, well known process steps have not been described in detail in order not to unnecessarily obscure the present invention.

Figures 1A and 1B were described in terms of the prior art. Figure 2 is a high-level illustration of an independent storage system 200, in accordance with an embodiment of the present invention. The independent storage system 200 includes a consumer computer 202 coupled to a network 204 via transport connection 210. Also included in the independent storage system 200 is a first and second independent storage node 206a and 206b, each coupled to the network 200 via an associated transport connection 210, and each coupled to an associated storage device 208a and 208b.

The network 204 can be any network or other transport capable of providing communication between computers, such LAN, or a WAN, such as the Internet, a bus, or wireless technology. The independent storage nodes 206a and 206b can be any type of node capable of receiving storage based commands over the network 204 and storing data to the associated storage device 208a and 208b. In one embodiment, the independent storage nodes 206a and 206b are nodes that include some intelligence, transport capability, I/O hardware capable of transmitting and receiving data via the transport connection 210 coupled the network 204, and software to perform the necessary file storage operations. The consumer computer 202 is typically a personal computer coupled to the network via a transport hardware, which a user uses to store and retrieve needed data.

The independent storage system 200 is capable of providing data storage to the consumer computer 202 remotely from the consumer computer 202 and independent of the particular operating system and file system executing on the consumer computer 202. In operation, a user utilizes the consumer computer 202 to run an application. When the user desires to store data, the user selects a destination for the data. The data is then transmitted from the consumer computer 202 though the transport connections 210 and network 204 to an independent storage node 206, such as independent storage node 206a, which then places the data on the associated storage device 208.

In one embodiment of the present invention, an application executing on the consumer computer 202 interacts with a storage device 208a of an independent storage node 206a as if the storage device 208a were local to the consumer computer 202. In this embodiment, the application executing on the consumer computer 202 has had the

transport specifics used to access the storage device 208a, and thus interacts with the storage device 208a as a traditional local drive. In this manner, the user does not need to learn a new file system to utilize the independent storage system 200.

Moreover, the independent storage system 200 can be used by multiple consumer computers 202, regardless of the specific OS executing on each of the consumer computers 202. Each consumer computer 202 can execute a different OS and file system, yet still share information stored on an independent storage node 206. Thus, a first consumer computer 202 executing a WINDOWS NT OS and a second consumer computer 202 executing a UNIX OS can both share the same file data on the storage device 208a of the independent storage node 206a. The storage node, 206a, presents directory and file information in the format that is native to the consumer computers' operating and file systems.

Figure 3 is a block diagram showing a remote independent storage system 300, in accordance with an embodiment of the present invention. The independent storage system 300 includes a consumer computer 202 and an independent storage node 206 coupled to the consumer computer 202 via a communication channel 320. As will be discussed in greater detail subsequently, the communication channel 320 instantiates communication between the consumer computer 202 and the independent storage node 206 through the transport 204, which may be a network, a bus, a wireless mechanism or other physical medium.

The consumer computer 202 includes an application 302, a file system 304, and an independent storage driver 306. The consumer computer 202 can be a personal computer, a server, a PDA or any other computing device having a processor, memory,

and transport hardware for communicating over the transport 204. As will be discussed later with reference to Figures 7A and 7B, embodiments of the present invention can also be used locally within a single invention.

The independent storage node 206 includes a file system translator 308, a dynamic flat file system 310, a device driver 312, and an I/O device 314 coupled to a storage device 208. The independent storage node 206 can be any type of node capable of receiving storage based commands over the network 204 and storing data to the associated storage device 208. In one embodiment, the independent storage node 206 is a hardware configuration including a processor, memory, and transport hardware capable of transmitting and receiving data via the transport connection coupled the network 204.

It should be noted that the independent storage node 206 is not generally restricted by hardware. Because of the modularity of the embodiments of the present invention, the independent storage node 206 of the embodiments of the present invention can operate in conjunction with many different types of processor, such as DSP or i960 chips.

Embodiments of the present invention make the storage device 208 appear local to the consumer computer 202. Specifically, the independent storage driver 306 appears to the file system 304 of the consumer computer 202 as a typical device driver. Actually, the independent storage driver 306 functions to communicate file system request from the file system 304 to the independent storage node 206 via the communication channel 320 created expressly for this purpose. Hence, the independent storage driver 306 intercepts file system request from the file system 304, and transmits the request to the independent storage node 206. In this manner, embodiments of the present invention can make the

storage device 208 appear as a local drive to the application 302 and file system 304 of the consumer computer 202.

In use, the application 302 transmits a file system request to the file system 304, which in turn passes the file system request to the independent storage driver 306. The independent storage driver 306 then transmits the file system request to the file system translator 308 of the independent storage node 206 via the communication channel 320.

The file system translator 308 configures the file system request into a second format based on the file system being executed on the independent storage node 206. For example, in the exemplary independent storage system 300 shown in Figure 3, the file system translator 308 configures the file system request into a second format based on a flat file system and provides the translated file system request to the dynamic flat file system 310, which provides the request the device driver 312. The device driver 312 converts the translated file system request into a typical block-level I/O request that is then passed from the I/O device 314 to the storage device 208.

Although the examples discussed herein use a flat file system on the independent storage node, it should be borne in mind that any file system can be used. When using a file system other than the flat file system discussed herein, the file system translator 308 configures the file system request into a format based on the file system executing on the independent storage node 206. It should be noted that the translator may be a no-op in some cases.

The dynamic flat file system 310 shown in Figure 3 is a file system constructed of volume-type, file-type and directory-type objects, structured as an array of objects. In one

embodiment, objects are of the class "volume", "directory" or of the class "file." Directory and volume objects reference other directories and files associated with the directories. All objects include an entry for their immediate parent object, which may be a node object, a volume object or a directory object.

The dynamic flat file system 310 incorporates object attributes that are a superset of other files system's attributes. In this manner, the dynamic flat file system 310 can represent files from other file systems whose attributes the dynamic flat file system 310 incorporates. As such, the dynamic flat file system 310 enables translation from the other file system formats to the dynamic flat file system 310 format, and vice versa.

The file system translator 308 translates from a disk-resident file system to a native file system format of the requester, and vice versa. To this end, the file system translator 308 combines a parsing and generation function with the ability to create file structures from more abstract attribute list, such as the attribute list of the dynamic flat file system 310. In use, the file system translator 308 reads the attributes for one file and maps those attributes to those that are native to the format that the consumer computer requires.

By using the file system translator 308 and the dynamic flat file system 310, the independent storage node 206 can perform file system request sent from the consumer computer 202 via the independent storage driver 306 regardless of the OS and file system executing on the consumer computer 202. Moreover, the independent storage driver 306 makes the storage device 208 appear local to the consumer computer 202.

Figure 4 is a block diagram showing an independent storage logical platform 400, in accordance with an embodiment of the present invention. The independent storage logical platform 400 include transport protocols module 402, a management function 404, file access privileges/security/file system translation module 406, a native file system/drivers module 408, and a hardware platform 410.

The independent storage logical platform 400 illustrates the modules used in embodiments of the present invention to make the independent storage system operate. The transport protocols module 402 and management module 404 are used to facilitate transfer of data and commands across the transport. In one embodiment, the transport protocols module 402 and management module 404 are part of the independent storage driver and reside on the consumer computer and the independent storage node. The privileges/security/file system translation module 406 is generally located on the independent storage node to facilitate data security and to translate file system request and data formats from the native file system of the independent storage node to the file system of the consumer computer.

The node-native file system module 408 typically resides on the independent storage node and facilitates access to the storage device driver. In one embodiment, the native file system module 408 includes the dynamic flat file system and associated device drivers. In addition, RAID and N-way Mirror software may be included in the native file system module 408. Finally, the hardware platform 410 represents the actual hardware that the above software modules are executed on.

Each module of the independent storage logical platform 400 includes a well-defined interface that allows dynamic binding of new modules. Further, the modular

design allows for platform-specific modules to be interchanged with other modules to facilitate platform independence. For example, an independent storage logical platform 400 created to operate over an Ethernet network can be reconfigured to operate on a Fibre Channel network by replacing the networking protocols module and NIC driver 402 without having to recreate an entire independent storage logical platform 400. Similarly, processor specific modules may be interchanged to allow an independent storage logical platform to operate on any microprocessor, regardless of the microprocessor architecture without modification of the source code.

In one embodiment of the present invention, the independent storage logical platform 400 is capable of communicating with a particular O/S environment using a Uniform Driver Interface (UDI) standard. The UDI standard provides a standard driver environment to an O/S. Specifically, the UDI accepts UDI interfaces and translates the interfaces into interfaces specific to the particular O/S to which the UDI was implemented. The UDI also accepts interface messages from the particular O/S and translates these into UDI interfaces usable by UDI based drivers.

Thus, embodiments of the present invention can be used with any O/S that provides a UDI environment without source code changes. Since many operating systems may provide a UDI environment, embodiments of the present invention can be incorporated into many different operating systems and the computer or other hardware platforms on which the OS is implemented.

Figure 5 is a block diagram showing a communication channel 320, in accordance with an embodiment of the present invention. The communication channel includes an independent storage driver 306, consumer protocol stacks 402a, consumer transport

hardware related driver 500a, a transport 210, independent storage node transport hardware related driver 500b, independent storage node protocol stacks 402b, and a file system translator 308.

The consumer and node transport hardware 500a and 500b are NICs that allow communication over the transport 210, such as Ethernet or Fibre Channel NICs. The consumer and node device drivers 500a and 500b are device drivers that control the related transport hardware, such as Ethernet drivers or Fibre Channel drivers. The transport 210 is any medium over which data can be communicated using the transport hardware, such as a twisted pair wire or wireless connection. Preferably, the transport 210 supports a transport protocol, such as Internet Protocol (IP), or if the transport is a bus, such as PCI, a bus protocol may be used. The transport 210 can also include a network, such as the Internet. Hence, the consumer side transport hardware 500a can be in communication with the node transport hardware 500b via the Internet.

The communication channel 320 uses messaging and/or memory semantics and is a logical connection that is independent of the underlying physical transport 210 and the transport's low-level protocols. The communication channel 320 abstracts the transport driver and transport hardware, and thus presents the application (through an independent storage driver) and storage (through a file system translator and drivers) with a logical direct communication link between the two. That is, communication channel 320 is hidden from upper level applications using the independent storage driver, such that the upper level applications are unaware of the communication channel 320. As will be discussed later, the communication channel 320 may completely reside on a single machine and may not require a NIC, per se. It should be noted that the upper level

applications interact with the storage device in essentially the same manner, whether the storage device is local or remote, since embodiments of the present invention make the storage device appear local to upper level applications.

Detailed examples of the transport portions of two communication channels illustrating the transport hardware abstraction achievable using a communication channel are shown next with reference to Figures 6A and 6B. Figure 6A is a block diagram showing an exemplary transport portion of a communication channel 600a, in accordance with an embodiment of the present invention. The exemplary transport portion of the communication channel 600a includes a first Ethernet driver 602a, first Ethernet hardware 604a, a transport 210, second Ethernet hardware 604b, and a second Ethernet driver 602b. In the exemplary transport portion of the communication channel 600a of Figure 6A, the first Ethernet driver 602a and hardware 604a reside on the consumer computer 202. Similarly, the second Ethernet driver 602b and hardware 604b reside on the independent storage node 206 in the example of Figure 6A.

Using the transport portion of the communication channel 600a, the consumer computer 202 can logically communicate with the independent storage node 206 using the first and second Ethernet drivers 602a and 602b. In this manner, the transport portion of the communication channel 600a forms part of an abstract logical connection between the consumer computer 202 and the independent storage node 206.

Figure 6B is a block diagram showing a transport portion of another exemplary communication channel 600b, in accordance with an embodiment of the present invention. The exemplary transport portion of the communication channel 600b includes an Ethernet driver 602a, Ethernet hardware 604a, a transport 210, an Ethernet/Fibre

Channel bridge 606, Fibre Channel hardware 608, and a Fibre Channel driver 610. The exemplary transport portion of the communication channel 600b illustrates one manner in which a communication channel 600b can be used as a logical communication mechanism that is independent of an underlying transport and its low-level protocols.

In use, data can travel from the Ethernet hardware 604a across the transport 210 to the Ethernet/Fibre Channel bridge 606. The Ethernet/Fibre Channel bridge 606 translates Ethernet commands into Fibre Channel commands, and vice versa, thus providing a mechanism to allow the Ethernet hardware 604a to communicate with the Fibre Channel hardware 608. The data then travels from the Ethernet/Fibre Channel bridge 606 to the Fibre Channel hardware 608 to the Fibre Channel driver 610. In this manner, the consumer computer 202 can communicate with the independent storage node 206 despite the difference in transport hardware of the two platforms. Thus, the transport portion of the communication channel 600b forms part of an abstract logical connection between the consumer computer 202 and the independent storage node 206.

As previously mentioned, the independent storage system of the embodiments of the present invention can be embodied on a single computer system. Figure 7A is a block diagram showing a local independent storage system 700, in accordance with an embodiment of the present invention. The local independent storage system 700 includes a consumer computer 202 coupled to a storage device 208 via a peripheral interconnect 702.

The consumer computer 202 includes an application program 302, a file system 304, and independent storage driver 306, and a file system translator 308 in communication with the independent storage driver 306 via a communication channel

320. The consumer computer 202 further includes a dynamic flat file system 310, a device driver 312, and an I/O device 314 coupled to the storage device 208 via the peripheral interconnect 702.

In use, the application 302 transmits a file system request to the file system 304, which in turn passes the file system request to the independent storage driver 306. The independent storage driver 306 then transmits the file system request to the file system translator 308 via the communication channel 320.

Although the storage device is local to the consumer computer 202, the local independent storage system 700 still uses a communication channel 320 to transmit file system request to the file system translator, and vice versa. In this case, the transport could be a Peripheral Component Interconnect (PCI) bus. By using the communication channel 320 as a logical connection, the local independent storage system 700 can be more easily configured to work with a remote independent storage system, such as shown in Figure 3, since the storage device 208 of both systems will appear local to the consumer computer 202. Thus, when the consumer computer 202 is accessing the storage device 208 of the local independent storage system 700, the communication channel of Figure 7A could be used. When the consumer computer 202 is accessing the storage device 208 of the remote independent storage system, the communication channel of Figure 3 could be used.

Referring back to Figure 7A, the file system translator 308 can configure the file system request into a second format based on the file system being used to storage data on the storage device 208. For example, in the exemplary local independent storage system 700 shown in Figure 7A, the file system translator 308 configures the file system request

into a second format based on a flat file system and provides the translated file system request to the dynamic flat file system 310, which provides the request the device driver 312. The device driver 312 converts the translated file system request into a block-level I/O request that is then passed from the I/O device 314 to the storage device 208. In another embodiment, the dynamic flat file system 310 replaces the native file system 304 of the consumer computer 202. In this embodiment, the file system translator 308 could be excluded from the system 700.

In some embodiments, the file system translation function 308 may not be required, because there may not be a need to support multiple file system types. In this case, the computer's 700 native file system 304 may be used and the communication channel 320 extends from the file system 306 to the storage device, as shown in Figure 7B. The communication channel 320 in this embodiment is comprised of objects representing a device driver, a bus driver, a local I/O bus and bus interface logic.

Figure 8 is a flowchart showing a method 800 for providing data storage independent of an underlying architecture, in accordance with an embodiment of the present invention. In a preprocess operation 802, preprocess operations are performed. Preprocess operations include mounting file systems and other preprocess operations that will be apparent to those skilled in the art.

In an intercepting operation 804, a file system request is intercepted. An application executing on the consumer computer generates a file system request, which is configured in the format of the O/S executing on the consumer computer. Once generated, the file system request is received by the independent storage driver, which appears to the file system of the consumer computer as a typical device driver. Actually,

the independent storage driver functions to communicate file system request from the file system to the independent storage node via the communication channel created expressly for this purpose.

The file system request is communicated to a file system translator, in a communication operation 806. The communication channel uses driver-to-driver communication semantics (messages or memory operations) and is a logical connection that is independent of the underlying physical transport and the transports low-level protocols. The communication channel abstracts the transport driver and transport hardware, and thus presents the independent storage driver and file system translator with a logical direct communication link between the two. Further, the communication channel is hidden from upper level applications using the independent storage driver, such that the upper level applications are unaware of the communication channel. As discussed previously, the communication channel may completely reside on a single machine. It should be noted that the upper level applications interact with the storage device in essentially the same manner, whether the storage device is local or remote, since embodiments of the present invention make the storage device appear local to upper level applications.

In a translation operation 808, the file system translator configures the file system request into a second format based on the O/S executing on the independent storage node. For example, the file system translator could configure the file system request into a second format based on a flat file system and provide the translated file system request to a dynamic flat file system executing on the independent storage node.

The translated file system request is then executed, in operation 810. Having translated and provided the file system request to the file system operating on the independent storage node, the file system provides the request to the device driver. The device driver converts the translated file system request into a block-level I/O request that is then passed from the I/O device to execute the request in conjunction with the storage device.

Post process operations are then performed in operation 812. Post process operations include communication channel maintenance, synchronization of file system data, and other post process operations that will be apparent to those skilled in the art. Advantageously, embodiments of the present invention provide platform independent storage for consumer computers, regardless of underlying hardware, operating system, or physical transport.

Figure 9 is block diagram showing a system 900 for independent storage communication, in accordance with an embodiment of the present invention. The system 900 includes a first independent storage node 206a and a second independent storage node 206b in communication with first independent storage node 206a via a communication channel 320, which may be a network or other technologies. The first independent storage node includes a dynamic flat file system 310a, a device driver 312a, and an I/O device 314a coupled to a storage device 208a. The second independent storage node includes a dynamic flat file system 310b, a device driver 312b, and an I/O device 314b coupled to a storage device 208b.

As shown in Figure 9, the first and second independent storage nodes 206a and 206b preferably utilize the dynamic flat file system 310. Using the communication

channel 320, the independent storage nodes 206a and 206b can perform execute a file consistency protocol that enables file sharing, locking, and n-way mirroring. As a result, a given consumer computer may share data with other consumer computers both through the locking and sharing mechanisms provided and through the File system translator. The embodiments of the present invention enable heterogeneous file systems to share data in this manner by providing for the translation of the dynamic flat file system to essentially any other file system format.

Figure 10 is a flowchart showing a method 1000 for providing independent storage using an independent storage node, in accordance with an embodiment of the present invention. The method 1000 illustrates the operation of a typical independent storage node. In an initial operation 1002, preprocess operations are performed. Preprocess operations include generating a file system request, transmitting the file system request, and other preprocess operations that will be apparent to those skilled in the art.

In a receiving operation 1004, the file system request is received from a requesting computer. Typically, the file system request is in a file system format specific to the requesting computer. In a translation operation 1006, the file system translator configures the file system request into a second format based on the O/S executing on the independent storage node. For example, the file system translator could configure the file system request into a second format based on a flat file system and provide the translated file system request to a dynamic flat file system executing on the independent storage node.

The translated file system request is then executed, in operation 1008. Having translated and provided the file system request to the file system operating on the independent storage node, the file system provides the request the device driver. The device driver converts the translated file system request into a block-level I/O request that is then passed from the I/O device to perform the request in conjunction with the storage device. Results of the performance of the file system request are then obtained and transmitted back to the requesting computer in a communication operation 1010.

Data is transferred per the request and the results obtained from performing the file system request are communicated to the requesting computer using a communication channel, in a communication operation 1010. As previously discussed, the communication channel uses driver-to-driver communication semantics and is a logical connection that is independent of the underlying physical transport and the transports low-level protocols. The communication channel abstracts the transport driver and transport hardware, and thus presents the independent storage driver and file system translator with a logical direct communication link between the two.

Further, the communication channel is hidden from upper level applications using the independent storage driver, such that the upper level applications are unaware of the communication channel specifics. As discussed above, the communication channel may completely reside on a single machine. It should be noted that the upper level applications interact with the storage device in essentially the same manner, whether the storage device is local or remote, since embodiments of the present invention make the storage device appear local to upper level applications.

Post process operations are then performed in operation 1012. Post process operations include communication channel maintenance and other post process operations that will be apparent to those skilled in the art. Advantageously, embodiments of the present invention provide platform independent storage for consumer computers, regardless of underlying hardware, operating system, or physical transport.

Although the foregoing invention has been described in some detail for purposes of clarity of understanding, it will be apparent that certain changes and modifications may be practiced within the scope of the appended claims. Accordingly, the present embodiments are to be considered as illustrative and not restrictive, and the invention is not to be limited to the details given herein, but may be modified within the scope and equivalents of the appended claims.

What is claimed is:

CLAIMS

1. A system for software module to module communication, comprising:

a module interface capable of receiving a message configured in a first format, the module interface further capable of translating the received message into a second format;

a first software module in communication with the module interface, the first software module capable of communicating messages configured in the first format to the module interface; and

a second software module in communication with the module interface, the second software module capable of communicating messages configured in the second format to the module interface, wherein the first software module is capable of communicating with the second software module via the module interface to facilitate data storage.

2. A system as recited in claim 1, wherein the module interface is further capable of translating the received message into a third format.

3. A system as recited in claim 2, wherein the second software module is capable of providing a first function related to a first hardware type.

4. A system as recited in claim 3, wherein a third software module capable of communicating messages configured in the third format to the module interface and capable of providing a second function related to a second hardware type can replace the second software module, and wherein the first software module is capable of communicating with the third software module via the module interface.

5. A system as recited in claim 5, wherein the first hardware type uses a SCSI protocol, and wherein the second hardware type uses a Fibre Channel protocol.

6. An independent storage node, comprising:

a processor;

transport hardware in communication with the processor, the transport hardware being capable of communicating data via a transport connection; and

modular storage software executing on the processor, the modular storage software comprising a plurality of software modules and a module interface that allows dynamic binding of the software modules, wherein the modular storage software is capable of executing on a plurality of processor types by using particular software modules related to a specific processor type.

7. An independent storage node as recited in claim 6, wherein the modular storage software is configured to execute on the specific processor type by replacing a particular software module included in the modular storage software with a new software module related to the specific processor type.

5

8. An independent storage node as recited in claim 6, wherein the new software module is capable of communicating with the processor via the module interface.

MODULAR SOFTWARE METHOD FOR INDEPENDENT STORAGE NODES

ABSTRACT OF THE DISCLOSURE

An independent storage node is disclosed that includes a processor and transport hardware in communication with the processor that is capable of communicating data via a transport connection. Executing on the processor is modular storage software that comprises a plurality of software modules and a module interface. The module interface allows dynamic binding of the software modules and is capable of executing on a plurality of processor types by using particular software modules related to a specific processor type.

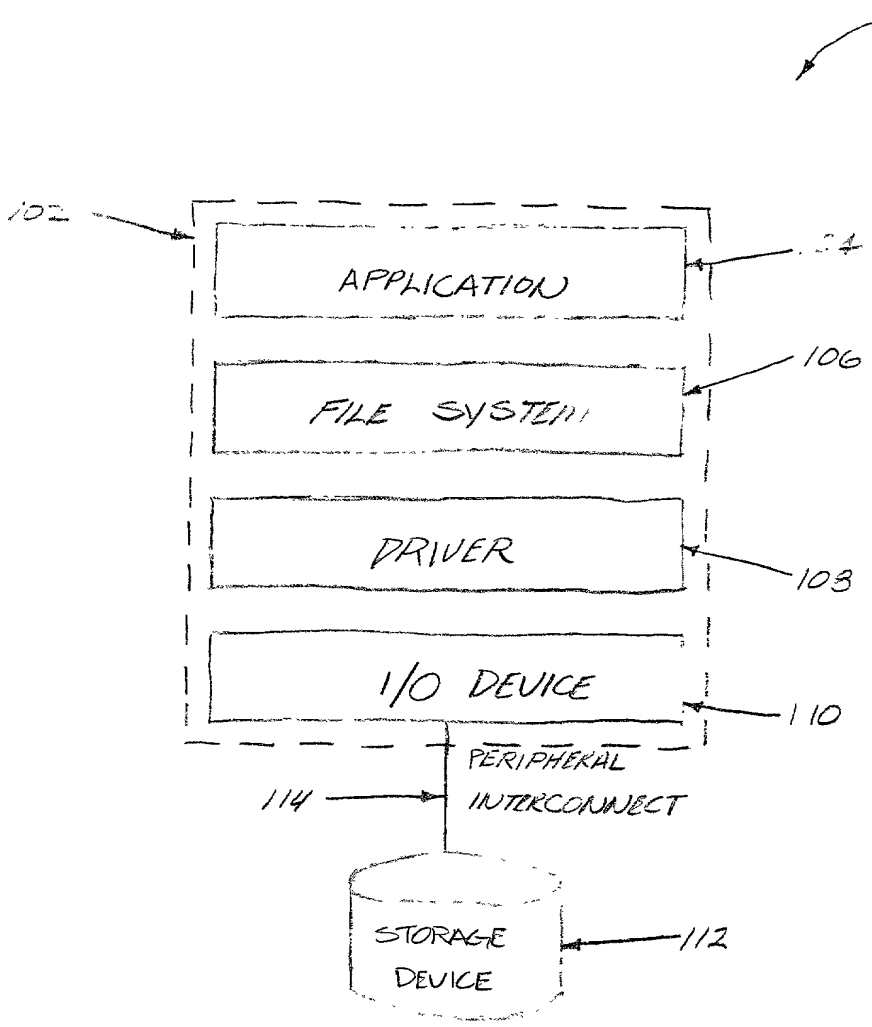


FIG. 1A
(PRIOR ART)

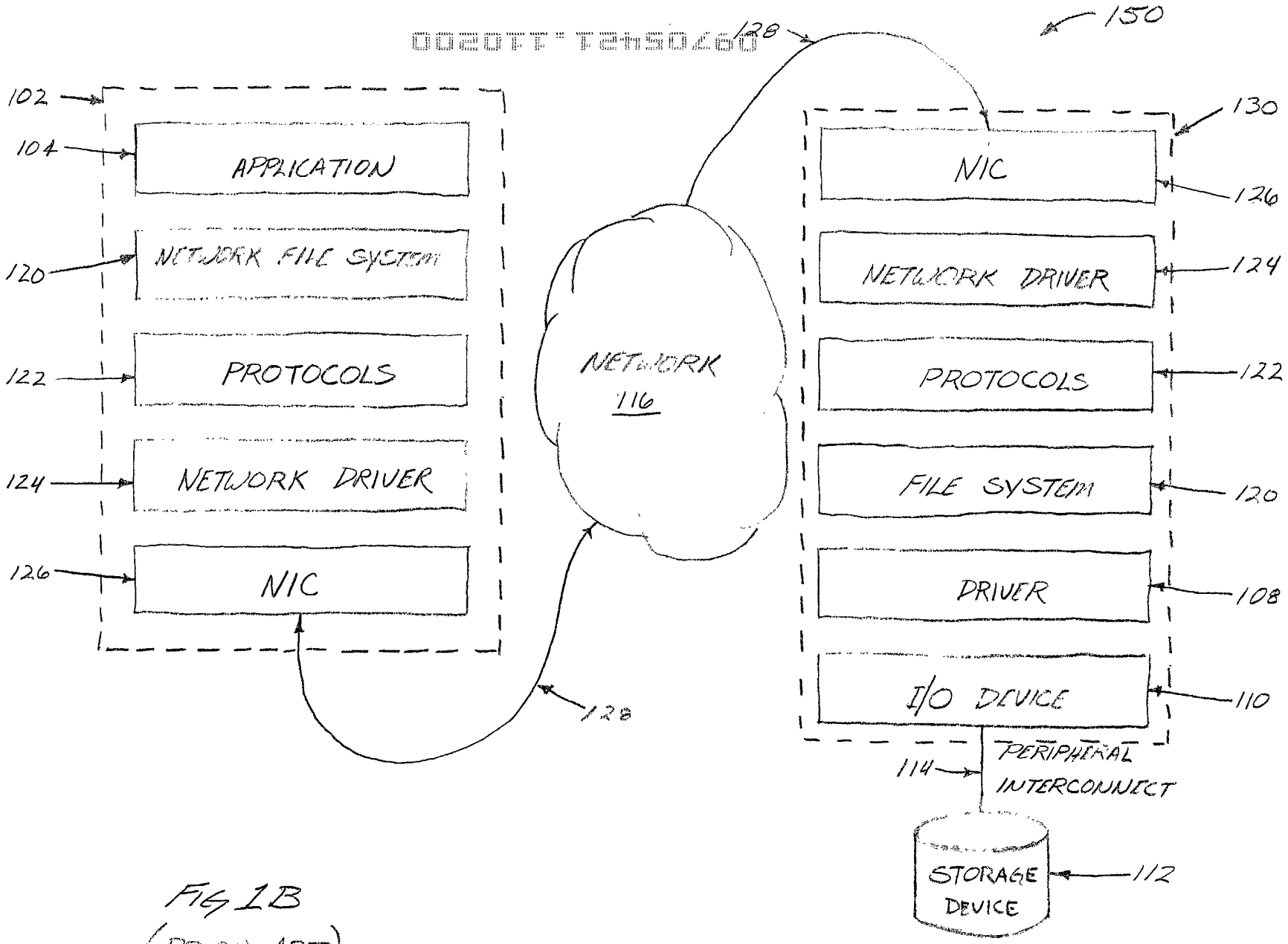


FIG 1B
(PRIOR ART)

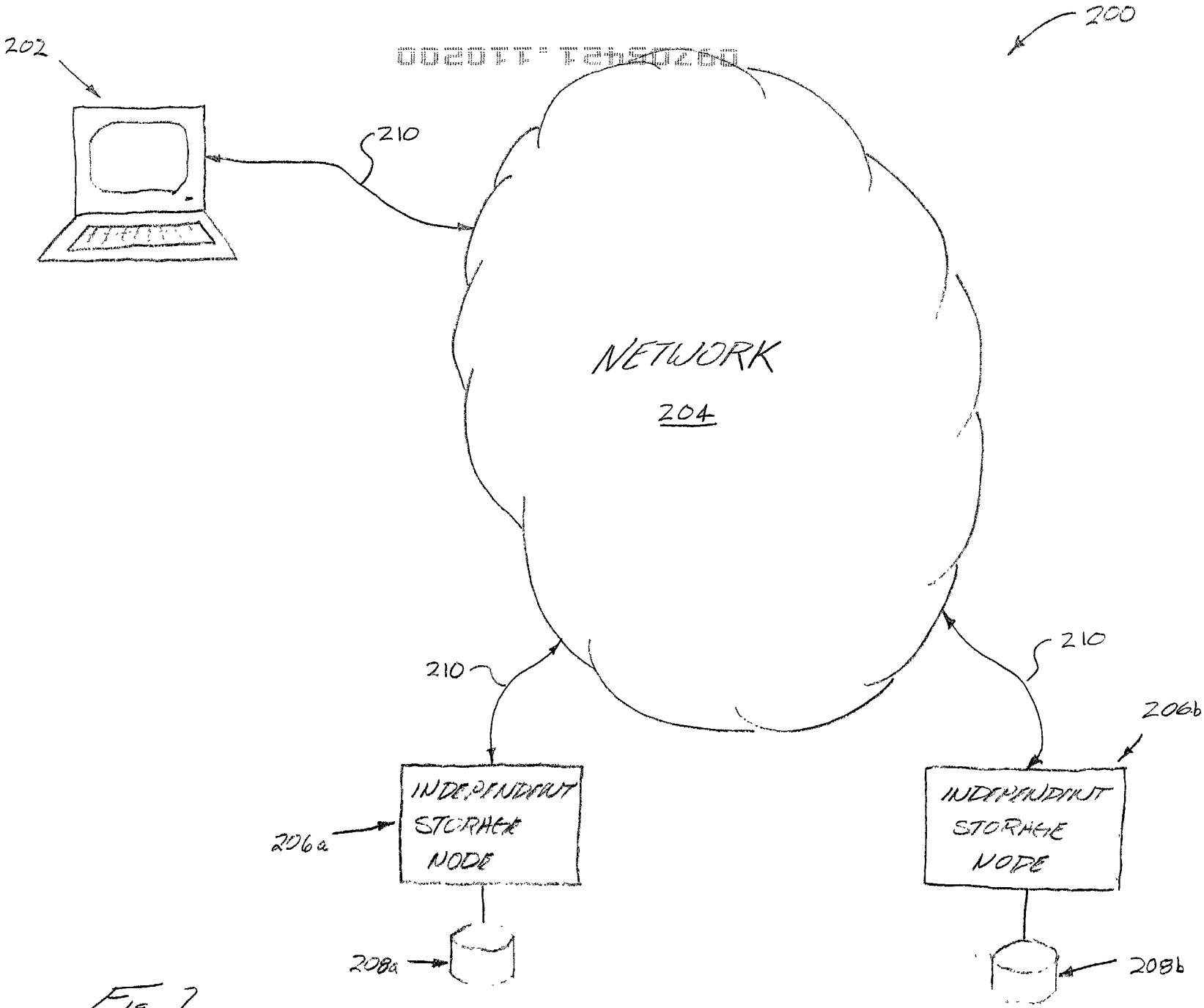

200
202
210
NETWORK
204
210
210
206b
INDEPENDENT
STORAGE
NODE
206a
INDEPENDENT
STORAGE
NODE
208a
208b

FIG 2

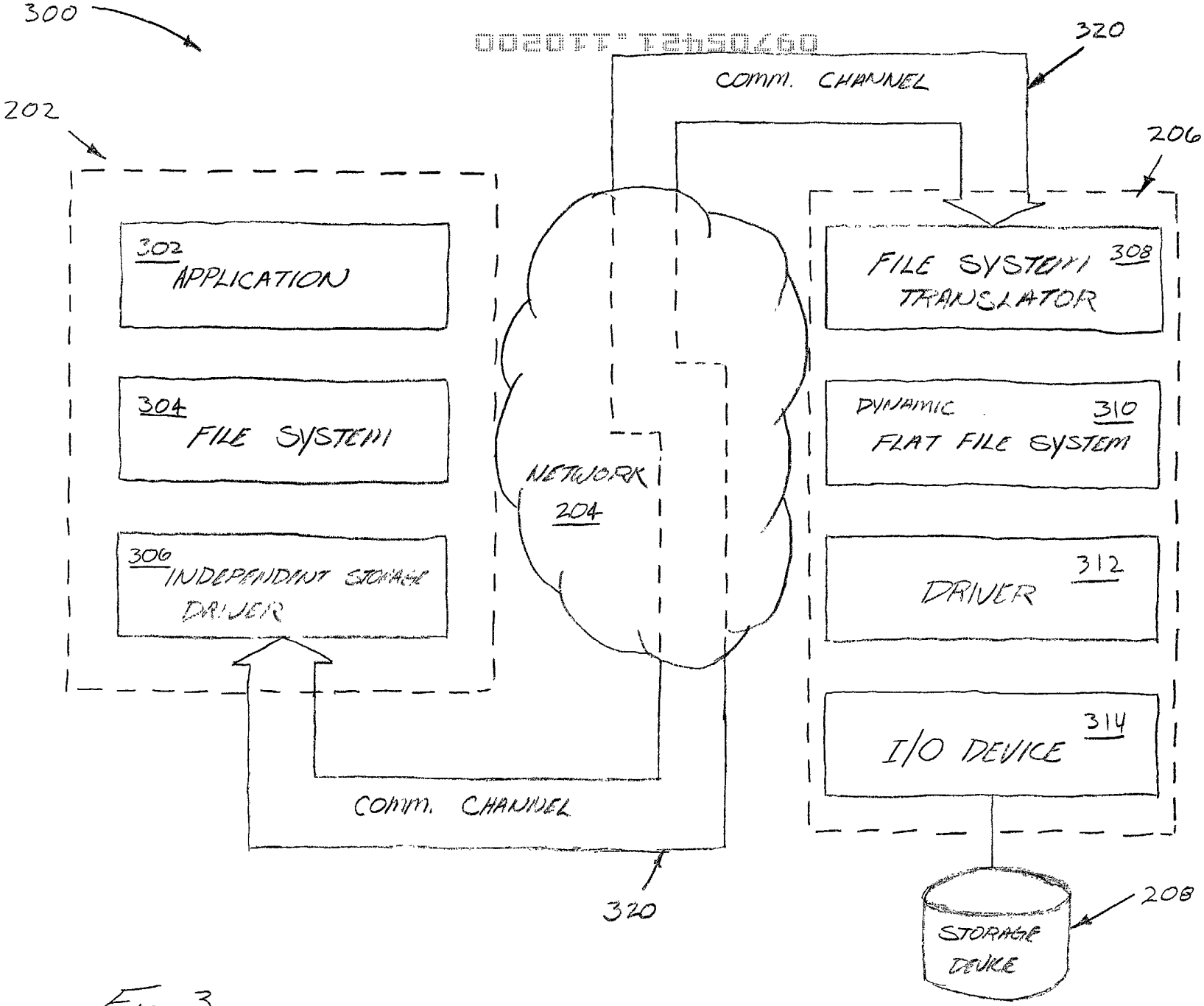

300
320
COMM. CHANNEL
202
206
302
APPLICATION
304
FILE SYSTEM
306
INDEPENDANT STORAGE DRIVER
NETWORK
204
FILE SYSTEM TRANSLATOR
308
DYNAMIC
310
FLAT FILE SYSTEM
312
DRIVER
314
I/O DEVICE
COMM. CHANNEL
320
208
STORAGE DEVICE

Fig 3

400

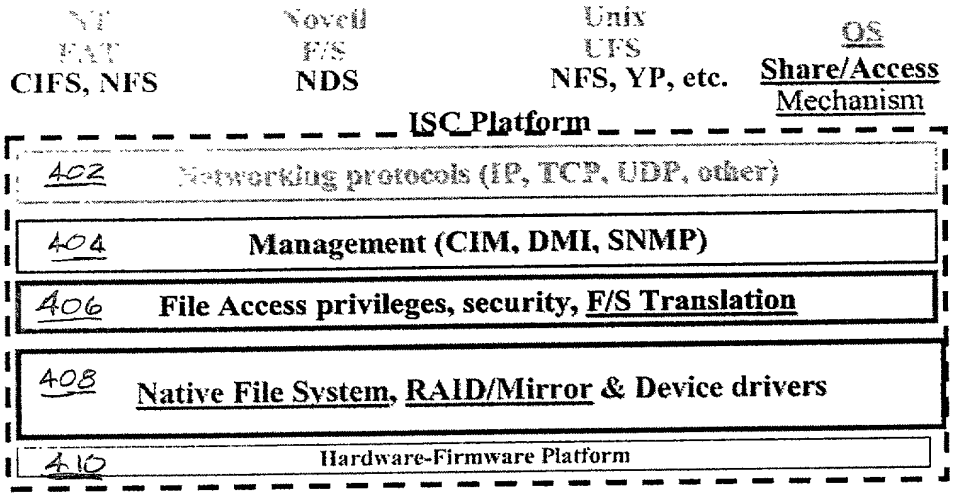
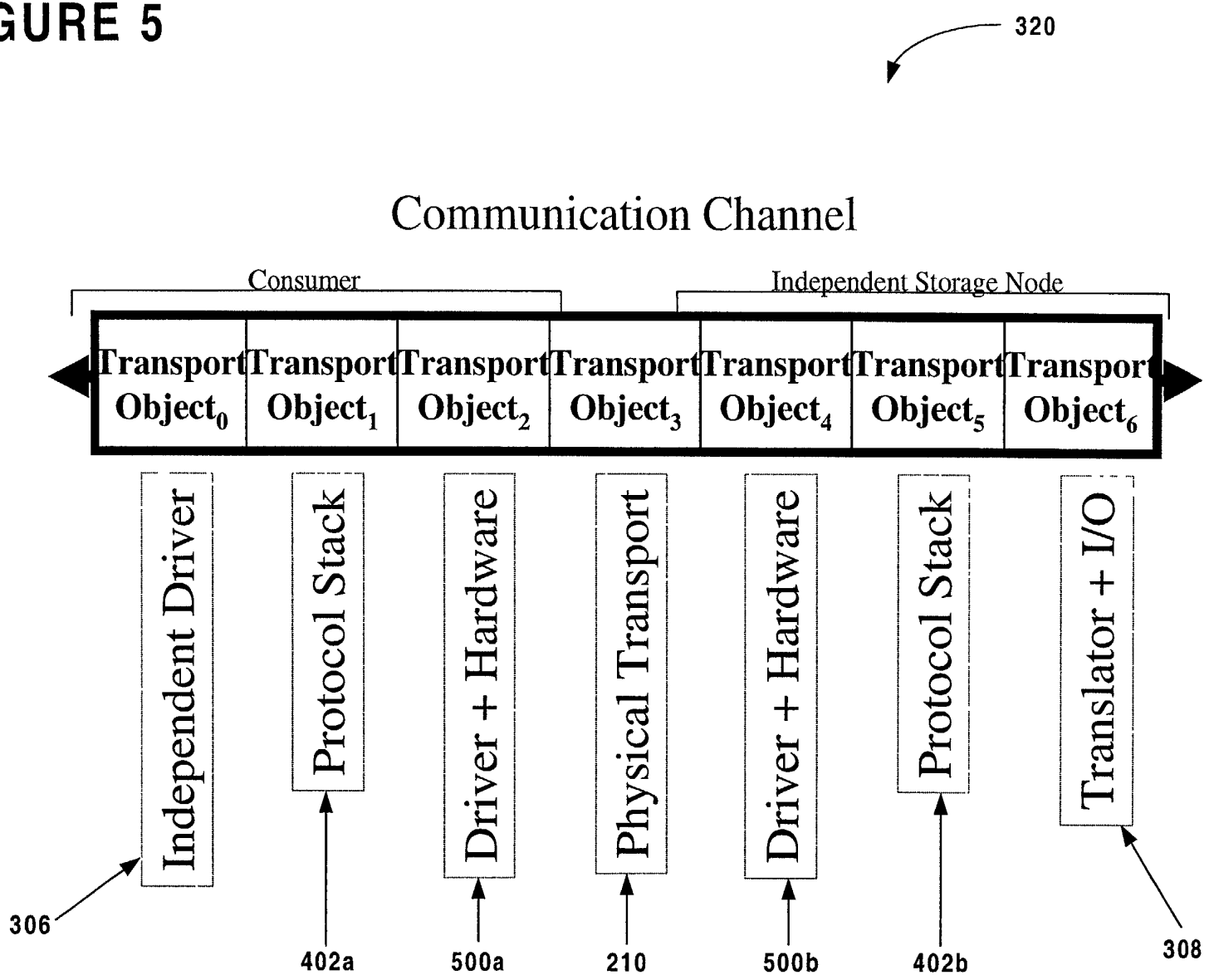


Fig 4

FIGURE 5



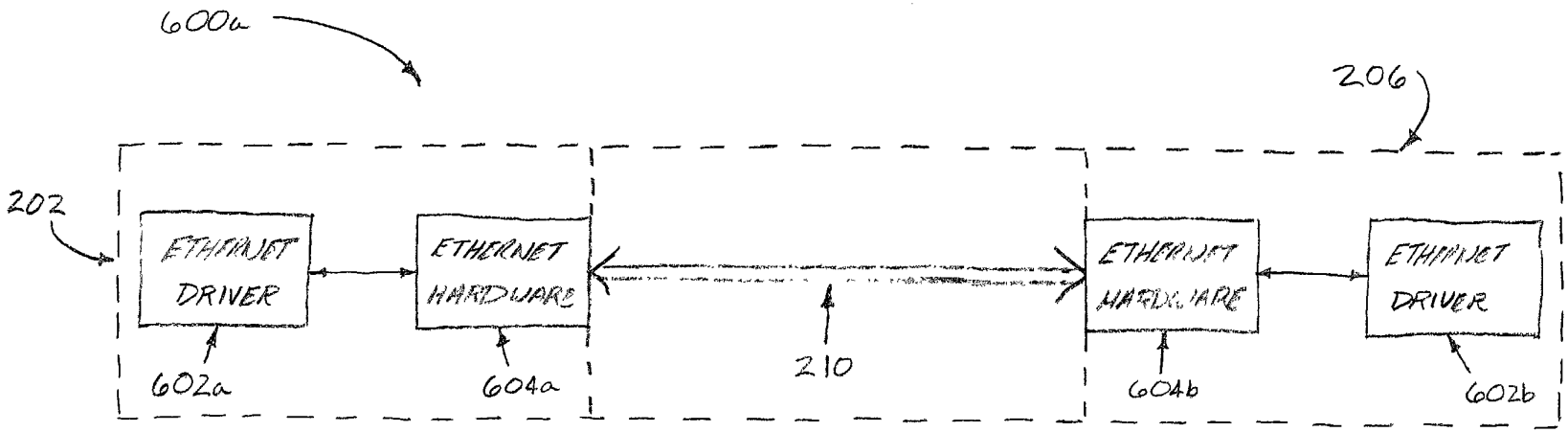


Fig 6A

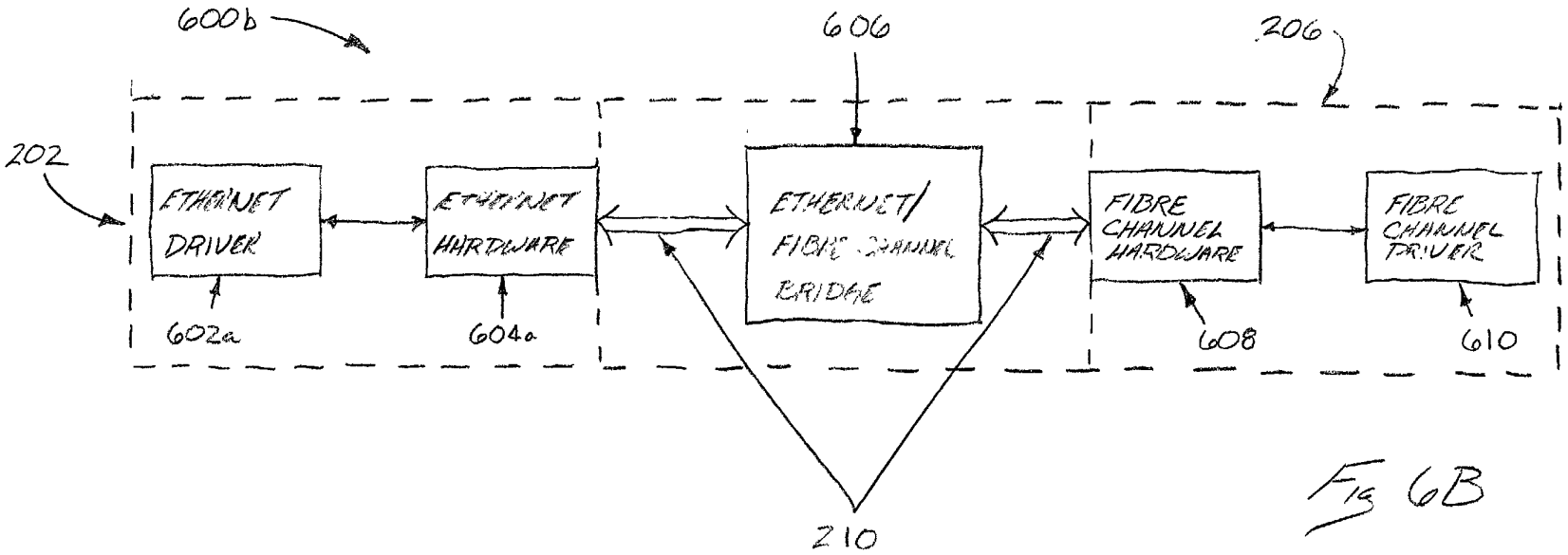


Fig 6B

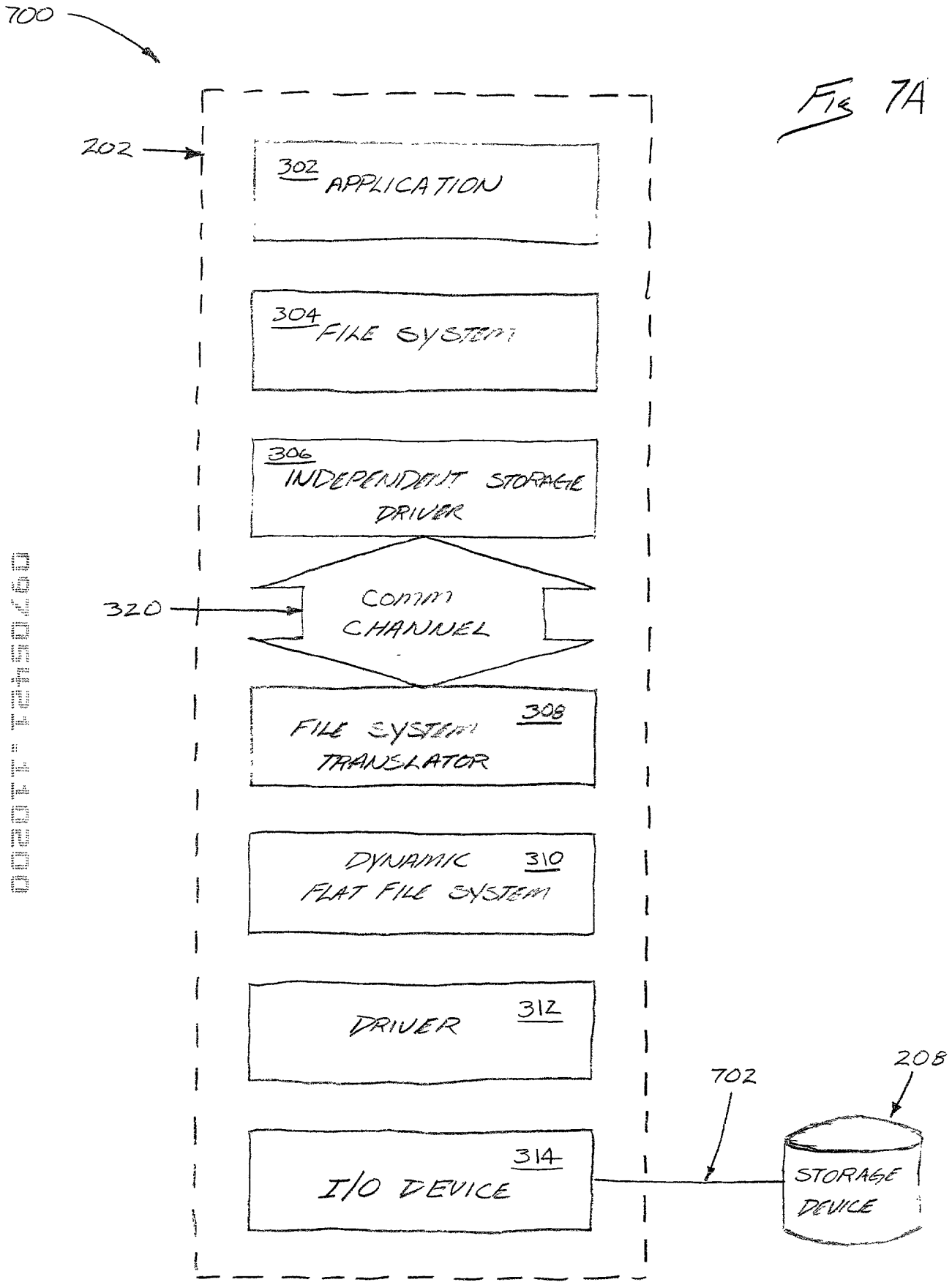

700
202
302 APPLICATION
304 FILE SYSTEM
306 INDEPENDENT STORAGE DRIVER
320
COMM CHANNEL
FILE SYSTEM TRANSLATOR 308
DYNAMIC FLAT FILE SYSTEM 310
DRIVER 312
I/O DEVICE 314
702
208
STORAGE DEVICE

Fig 7A

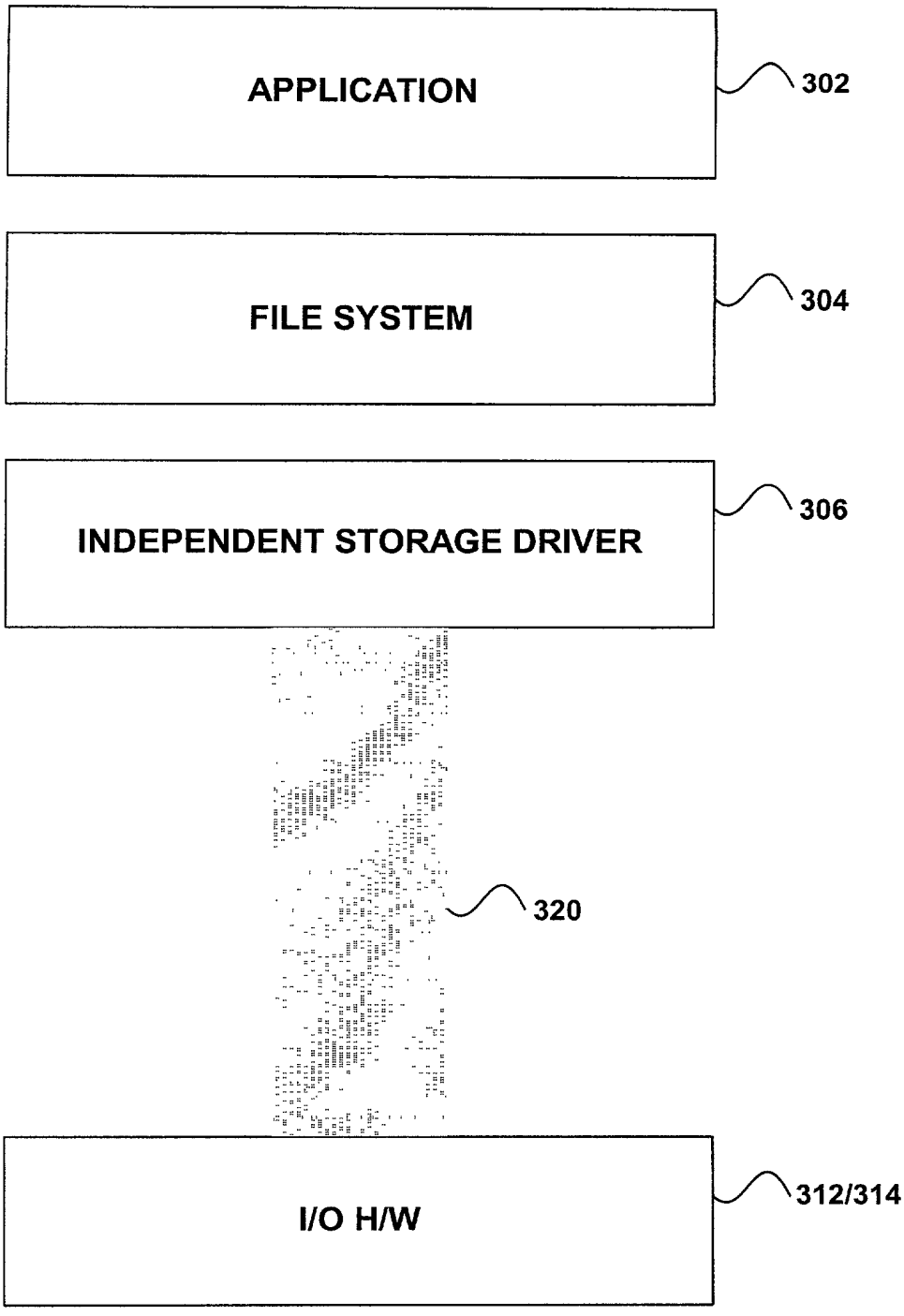

APPLICATION
302
FILE SYSTEM
304
INDEPENDENT STORAGE DRIVER
306
320
I/O H/W
312/314

FIGURE 7B

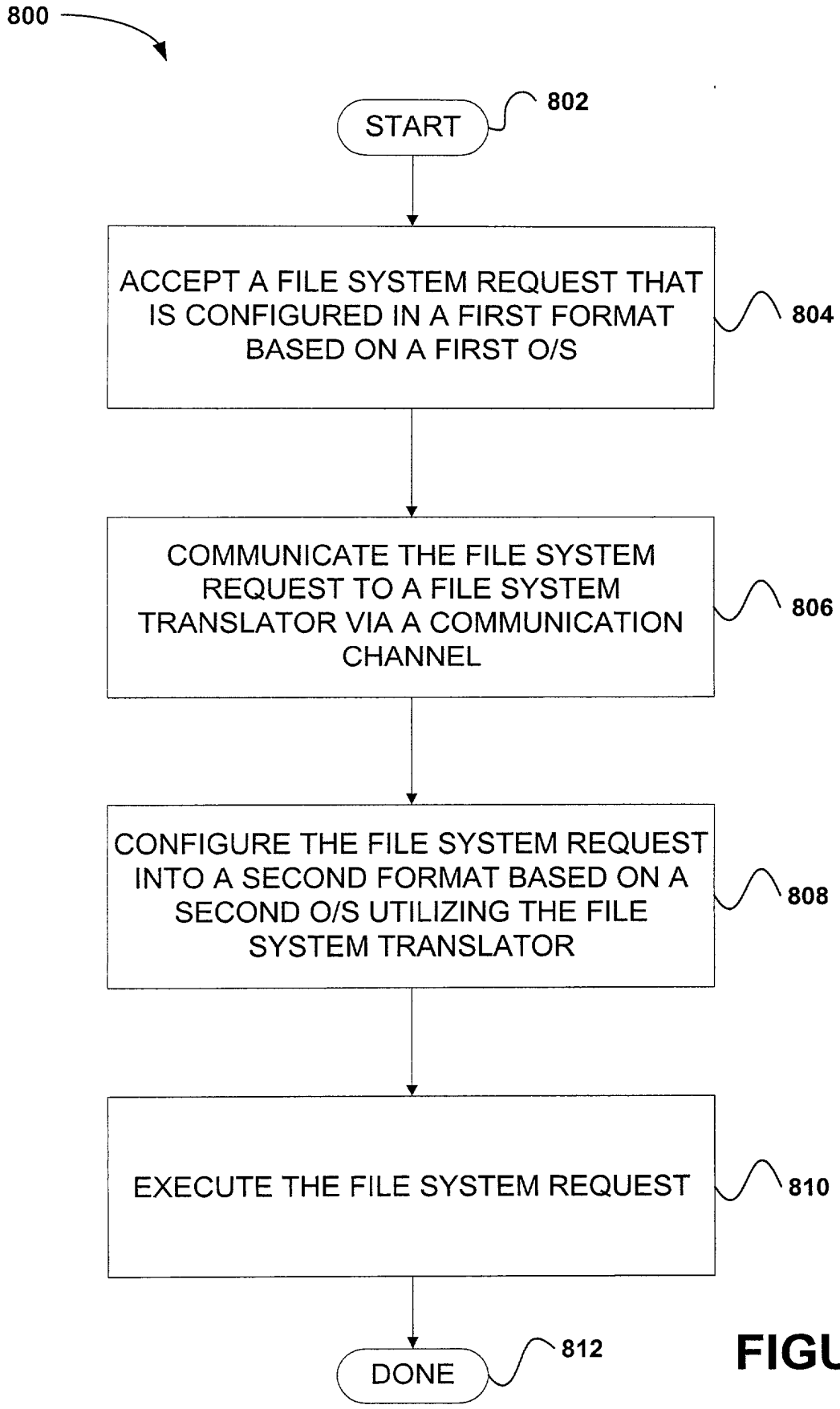

800
START
802
ACCEPT A FILE SYSTEM REQUEST THAT IS CONFIGURED IN A FIRST FORMAT BASED ON A FIRST O/S
804
COMMUNICATE THE FILE SYSTEM REQUEST TO A FILE SYSTEM TRANSLATOR VIA A COMMUNICATION CHANNEL
806
CONFIGURE THE FILE SYSTEM REQUEST INTO A SECOND FORMAT BASED ON A SECOND O/S UTILIZING THE FILE SYSTEM TRANSLATOR
808
EXECUTE THE FILE SYSTEM REQUEST
810
DONE
812

FIGURE 8

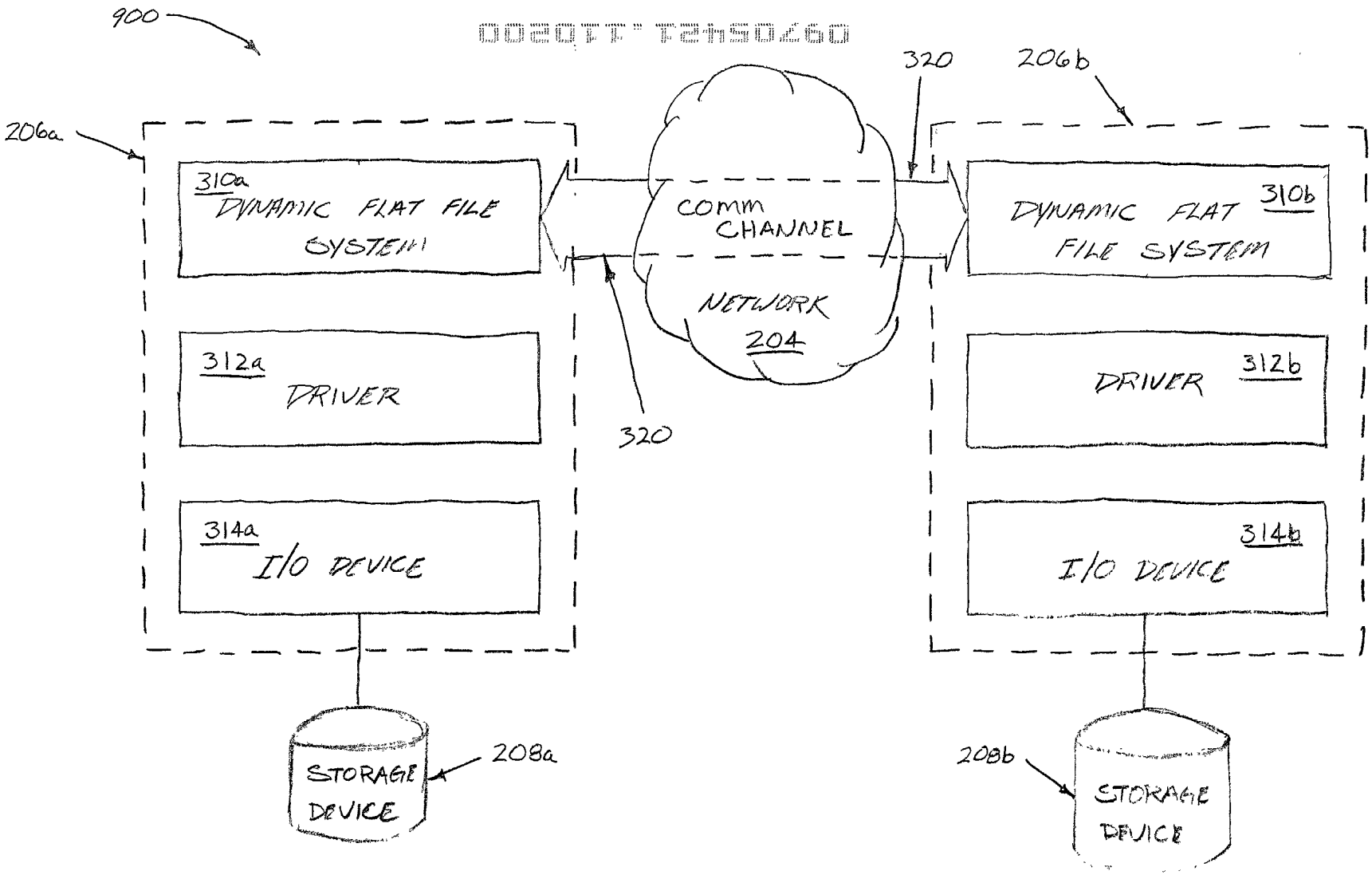

900
320
206b
206a
310a
DYNAMIC FLAT FILE SYSTEM
COMM CHANNEL
NETWORK
204
310b
DYNAMIC FLAT FILE SYSTEM
312a
DRIVER
320
312b
DRIVER
314a
I/O DEVICE
314b
I/O DEVICE
208a
STORAGE DEVICE
208b
STORAGE DEVICE

Fig 9

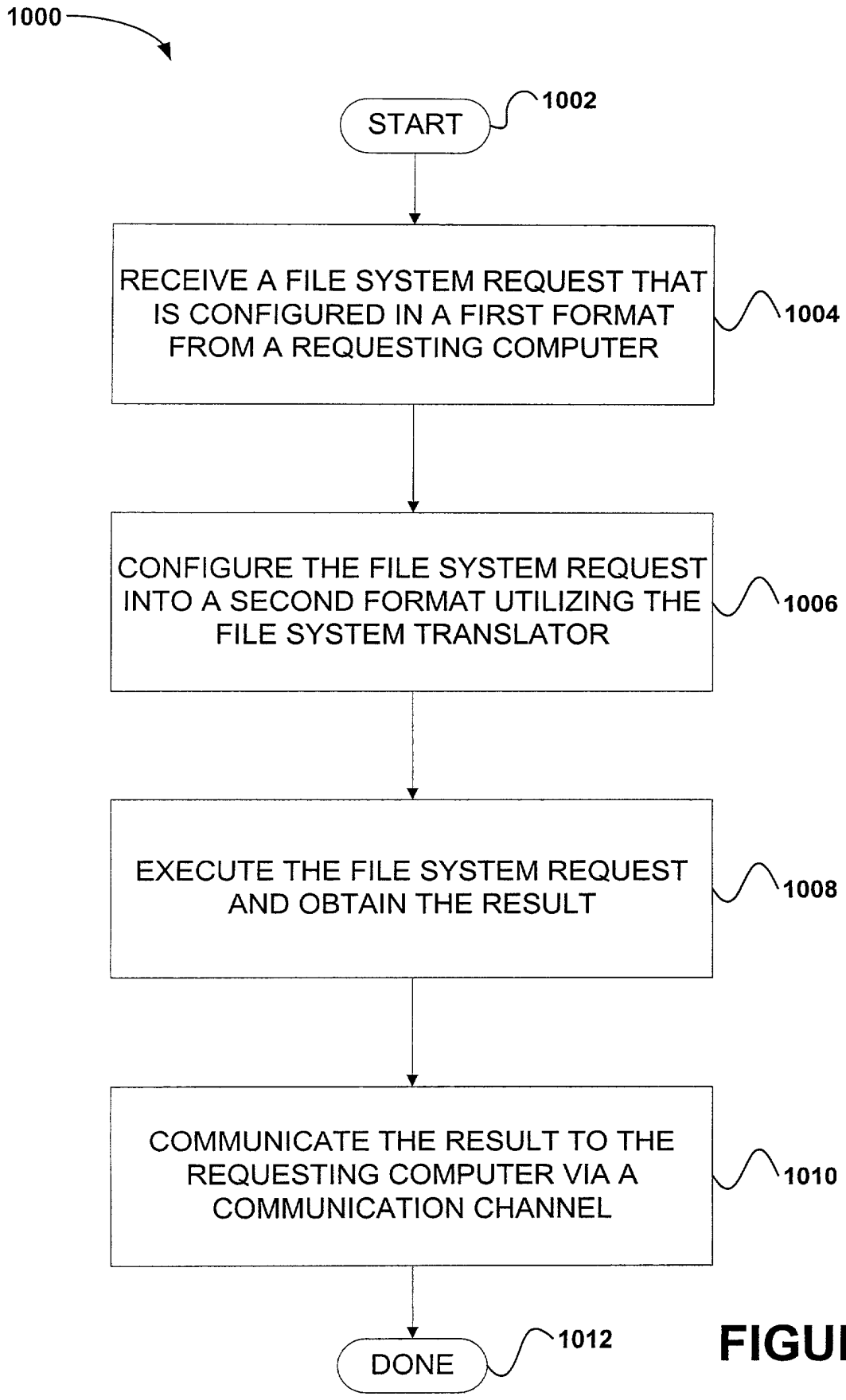

1000
START
1002
RECEIVE A FILE SYSTEM REQUEST THAT IS CONFIGURED IN A FIRST FORMAT FROM A REQUESTING COMPUTER
1004
CONFIGURE THE FILE SYSTEM REQUEST INTO A SECOND FORMAT UTILIZING THE FILE SYSTEM TRANSLATOR
1006
EXECUTE THE FILE SYSTEM REQUEST AND OBTAIN THE RESULT
1008
COMMUNICATE THE RESULT TO THE REQUESTING COMPUTER VIA A COMMUNICATION CHANNEL
1010
DONE
1012

FIGURE 10

DECLARATION AND POWER OF ATTORNEY FOR ORIGINAL U.S. PATENT APPLICATION

Attorney's Docket No. INSTP007B

As a below-named inventor, I hereby declare that:

My residence, post office address and citizenship are as stated below next to my name.

I believe that I am the original, first and sole inventor (if only one name is listed below) or an original, first and joint inventor (if plural names are listed below) of the subject matter which is claimed and for which a patent is sought on the invention entitled: MODULAR SOFTWARE METHOD FOR INDEPENDENT STORAGE NODES, the specification of which,

(check one)

1. ☒ is attached hereto.

2. ☐ was filed on _____ as U.S. Application Serial No. _____ and was amended on _____.

3. ☐ was filed on _____ as International PCT Application Serial No. _____ and was amended on _____.

I hereby state that I have reviewed and understand the contents of the above-identified specification, including the claims, as amended by any amendment referred to above.

I acknowledge the duty to disclose information which is material to the examination of this application in accordance with Title 37, CFR § 1.56.

I hereby claim foreign priority benefits under Title 35, United States code, § 119(a)-(d) or § 365(b) of any foreign application(s) for patent or inventor's certificate, or § 365(a) of any PCT International application which designated at least one country other than the United States, listed below and have identified below, by checking the box, any foreign application for patent or inventor's certificate, or PCT International application having a filing date before that of the application on which priority is claimed:

Prior Foreign Application(s)			Priority Benefits Claimed?
_____ (Appl. No.)	_____ (Country)	_____ (Filing Date)	☐ Yes ☐ No
_____ (Appl. No.)	_____ (Country)	_____ (Filing Date)	☐ Yes ☐ No
_____ (Appl. No.)	_____ (Country)	_____ (Filing Date)	☐ Yes ☐ No

I hereby claim the benefit under 35 U.S.C. §119(e) of any United States provisional application(s) listed below:

_____ (Application Serial No.) _____ (Filing Date)

_____ (Application Serial No.) _____ (Filing Date)

I hereby claim the benefit under Title 35, United States Code, § 120 of any United States application(s), or § 365(c) of any PCT International application designating the United States, listed below and, insofar as the subject matter of each of the claims of this application is not disclosed in the prior United States or PCT International application in the manner provided by the first paragraph of Title 35, United States Code, § 112, I acknowledge the duty to disclose information which is material to patentability as defined in Title 37, Code of Federal Regulations, § 1.56 which became available between the filing date of the prior application and the national or PCT international filing date of this application:

Prior U.S. Application(s)

(Application Serial No.)	(Filing Date)	(Status - patented, pending, abandoned)
(Application Serial No.)	(Filing Date)	(Status - patented, pending, abandoned)

And I hereby appoint the law firm of Martine Penilla & Kim, LLP, including **Peter B. Martine (Reg. No. 32,043); Albert S. Penilla (Reg. No. 39,487); Raymis H. Kim (Reg. No. 39,462); Chester E. Martine, Jr. (Reg. No. 19,711); Edmund H. Mizumoto (Reg. No. 46,938); and Joe A. Brock II (Reg. No. 46,021),** as my principal attorneys to prosecute this application and to transact all business in the Patent and Trademark Office connected therewith:

Send Correspondence To:

Joe A. Brock
MARTINE PENILLA & KIM, LLP
710 Lakeway Drive, Suite 170
Sunnyvale, CA 94085

Direct Telephone Calls To: **Joe A. Brock II at telephone number (408) 749-6900, ext. 6920**

I hereby declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true; and further that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under section 1001 of Title 18 of the United States Code, and that such willful false statements may jeopardize the validity of the application or any patent issuing thereon.

Typewritten Full Name of Sole or First Inventor: Mark W. Bradley Citizenship: USA

Inventor's signature: Mark W. Bradley **Date of Signature:** 11/2/2000

Residence: (City) Boulder (State/Country) CO

Post Office Address: 1946 Lefthand Canyon Drive, Boulder, CO 80302